Example 11. Determine $16^{-1} \pmod{25}$.

Solution. We apply the extended Euclidean algorithm:

$$\begin{aligned} &\gcd(16,25) && \boxed{25} = 2\cdot\boxed{16} - 7 && \text{or:} && \boxed{A} \quad 7 = -1\cdot\boxed{25} + 2\cdot\boxed{16} \\ &= \gcd(7,16) && \boxed{16} = 2\cdot\boxed{7} + 2 && && \boxed{B} \quad 2 = 1\cdot\boxed{16} - 2\cdot\boxed{7} \\ &= \gcd(2,7) && \boxed{7} = 3\cdot\boxed{2} + 1 && && \boxed{C} \quad 1 = \boxed{7} - 3\cdot\boxed{2} \\ &= 1 \end{aligned}$$

Backtracking through this, we find that:

$$1 \underset{\boxed{C}}{=} \boxed{7} - 3\cdot\boxed{2} \underset{\boxed{B}}{=} 7\cdot\boxed{7} - 3\cdot\boxed{16} \underset{\boxed{A}}{=} -7\cdot\boxed{25} + 11\cdot\boxed{16}$$

That is, **Bézout's identity** takes the form $-7\cdot 25 + 11\cdot 16 = 1$.
Reducing modulo 25, we get $11\cdot 16 \equiv 1 \pmod{25}$. Hence, $16^{-1} \equiv 11 \pmod{25}$.

Application: credit card numbers

Have you ever thought about the numbers on your credit card? Usually, these are 16 digits. For instance, 4266 8342 8412 9270.

No worries (or false hopes...). While close, this is not exactly my credit card number.

- The first digit(s) of a credit card identify the issuer of the card. For instance, a leading 4 is typically Visa, 51 to 55 indicate Mastercard, and 34, 37 indicate American Express. The above credit card is indeed a Visa card.

  More information at: https://en.wikipedia.org/wiki/Payment_card_number

- The last digit is a **check digit**, and a valid credit card number must pass the **Luhn check** (patented by IBM scientist Hans Peter Luhn in 1954/60; now in public domain).

  This works as follows: every second digit, starting with the first, is doubled. If that results in a two-digit number, we take the sum of those two digits.

$$\begin{bmatrix} & 4 & 2 & 6 & 6 & 8 & 3 & 4 & 2 & 8 & 4 & 1 & 2 & 9 & 2 & 7 & 0 \\ \times 2 & 8 & & 12 & & 16 & & 8 & & 16 & & 2 & & 18 & & 14 & \\ & 8 & 2 & 3 & 6 & 7 & 3 & 8 & 2 & 7 & 4 & 2 & 2 & 9 & 2 & 5 & 0 \end{bmatrix}$$

  The other half of the digits is left unchanged. We then add all these digits and reduce modulo 10:

$$8+2+3+6+7+3+8+2+7+4+2+2+9+2+5+0 \equiv 0 \pmod{10}$$

  The result of that computation must be 0. Otherwise, the credit card number fails the Luhn check and is invalid.

Example 12. (extra exercise)

(a) Check that the number 4266 8342 8412 9280 fails the Luhn check.

(b) How do we have to change the last digit to turn this into a valid credit card number?

Armin Straub
straub@southalabama.edu

The purpose of the Luhn check is to detect accidental errors.

[A random credit card number has a 90% chance of failing the Luhn check. Why?!]

On the other hand, as the previous example shows, it provides basically no protection against malicious attacks (except against amateur criminals not aware of the Luhn check).

The Luhn check was designed before online banking (patent filed in 1954). So a special focus is on detecting accidental errors that occur frequently when writing down (things like) credit card numbers by hand.

- For instance, it is common that a single digit gets messed up. Every such error is detected by the Luhn check. (Why?!)
- Another common error is to transpose two digits. Every such error (with the exception of 09 versus 90) is detected.

  For instance. A 82 at the beginning contributes $7+2=9$ to the check sum, while a 28 contributes $4+8\equiv 2$ to the sum. Hence, replacing one with the other will result in the Luhn check failing.

  Advanced comment. An alternative checksum formula that can detect all single digit changes as well as all transpositions is the Verhoeff algorithm (1969). It is, however, much more complicated and cannot be readily performed by hand.

Example 13. The doubling and sum-of-digits procedure permutes the digits as follows:

original digit	0	1	2	3	4	5	6	7	8	9
adjusted digit	0	2	4	6	8	1	3	5	7	9
difference $(\mathrm{mod}\,10)$	0	1	2	3	4	6	7	8	9	0

Note. Looking at the differences modulo 10, we can see why the Luhn check is able to detect all transposition errors (except 09 versus 90).

Example 14. The Luhn check has the somewhat complicated feature that every second digit has to be doubled. Why do we not just add all the original digits and reduce the sum modulo 10?

Solution. One reason is that this simplified check does not catch the transposition of two digits. Why?!

[On the other hand, that simplified check does also detect if just a single digit is incorrect.]

Example 15. (extra) The International Standard Book Number ISBN-10 consists of nine digits $a_1a_2...a_9$ followed by a tenth check digit a_{10} (the symbol X is used if the digit equals 10), which satisfies

$$a_{10}\equiv\sum_{k=1}^{9} ka_k \pmod{11}.$$

The ISBN 0-13-186239-? is missing the check digit (printed as "?"). Compute it!

Solution. $1\cdot 0+2\cdot 1+3\cdot 3+4\cdot 1+5\cdot 8+6\cdot 6+7\cdot 2+8\cdot 3+9\cdot 9=210\equiv 1 \pmod{11}$

Hence, the full ISBN is 0-13-186239-1.

This is another example of **error checking**, which is standard practice for all sorts of identification numbers (such as bank account numbers, VIN). With a little more effort **error correction** is also possible.

Comment. The check digit is designed so that it is always possible to detect when a single digit is messed up. It is also always possible to detect when two digits are transposed.